REMARKS

Claims 1-2, 4-5, 12-16 and 19-20 were pending in the application. By this amendment, Applicants add dependent Claim 21. No new matter is added by the amendment. Applicants believe that no additional filing fee is due for the addition of one new dependent claim, wherein the total number of claims does not exceed 20 total claims.

The Examiner has rejected Claims 1, 2 and 15-16 under 35 USC 103(a) as unpatentable over van der Made in view of Kilpatrick and Kolichtchak; Claims 4, 5 and 12 as unpatentable over Hoefelmeyer in view of Kephart; Claims 13 and 14 as unpatentable over Hoefelmeyer in view of Kephart and Lamburt; and Claims 19-20 as unpatentable over van der Made in view of Kilpatrick, Kolichtchak, Hoefelmeyer and Lamburt. For the reasons set forth below, Applicants believe that the claims are patentable over the cited art.

The present invention provides a method, program storage device and apparatus for preventing attacks in data processing systems. The intrusion detection system comprises a host or application based sensor for detecting code based intrusions with a relatively low false-positive rate by monitoring system calls. Further, the present invention isolates the malicious code in the stack in order

to extract it before the malicious code can do damage to the system, as expressly recited in independent Claims 1, 15 and 16. Malicious code strings related to a detected intrusion are identified, extracted and forwarded to a pattern filter located in the monitored data processing system to prevent further intrusions using said malicious code strings. The malicious code strings may be forwarded to a response server for assembling sets of similar malicious code strings for which signatures are generated to permit identification of all malicious code strings contained in a set. The generated signatures are then distributed to monitored and/or monitoring systems of a protected network to prevent further intrusions using the malicious code strings and variations thereof. The generated signatures of the present invention are not simply the malicious code strings, but are signatures generated for correlated sets of code strings and patterns, as set forth on page 7, lines 9-21. The steps and means for correlating the malicious code strings and generating a signature that can be sent to other entities to facilitate identification and isolation of malicious code are recited in Claims 12, 19, 20 and 21.

Claims 1, 2, 15 and 16 have been rejected as unpatentable over van der Made in view of Kilpatrick and

Kolichtchak. The van der Made patent is directed to detecting malicious code. However, what van der Made provides is a virtual machine which simulates the functionality of the central processing unit and executes received code to determine if the received code is malicious. The virtual machine runs the code, generates a behavior pattern of the executed received code, and analyzes the behavior patterns to determine if they are characteristic of an intrusion. The van der Made method fully executes the code in a simulation environment, thereby protecting the CPU. As such, van der Made's approach does not have system calls and does not monitor system calls.

Applicants respectfully assert that van der Made does not detect intrusion by monitoring system calls to identify the intrusive code before it executes; rather, van der Made fully executes the intrusive code and identifies the code as malicious once it sees the full behavior pattern. Further, once van der Made detects a behavior pattern that is associated with malicious code, based on comparison to known behavior patterns of known malicious code, the virtual machine is shut down and its memory resources are released (Col. 9, lines 25-33). Accordingly, the code cannot execute in the actual computing system. The van der

Made patent does not teach or suggest extracting the malicious code string or generating a signature for forwarding to an intrusion limitation system.

Clearly van der Made does not identify malicious code strings based on monitored system calls, since the program is not actually executing in the system, but in a virtual machine. Moreover, van der Made does not teach or suggest matching system calls with established patterns and rules wherein the matching comprises establishing a non-deterministic automaton based on an analysis of executable code of a daemon executing in memory. The Examiner states that van der Made teaches the foregoing yet fails to point to any van der Made passages which teach or suggest the foregoing claim feature. The Examiner cites the van der Made Abstract and one van der Made passage from Col. 2, line 50 to Col. 3, line 15 against all of the claim features, without pointing to specific teachings and pointing out how those specific teachings relate to the pending claim language. The cited passage makes no mention of matching, code analysis, establishing an automaton, extracting malicious code by locating it in the stack (further discussed below with reference to the Kolichtchak patent) or forwarding extracted code strings to an intrusion limitation subsystem. Applicants acknowledge

that the van der Made Abstract does teach storing behavior patterns, but argue that the claim language recites far more than merely storing the behavior pattern of a program executed in a virtual machine.

It is well settled in U. S. Patent Law that "rejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness." In re Kahn, 441 F.3d 977, 988, 78 USPQ2d 1329, 1336, quoted with approval in KSR Int'l Co. v. Teleflex Inc., 127 S. Ct. 1727, 1741, 82 USPQ2d 1385, 1396 (2007). The Examiner's conclusions with respect to the Abstract and the one repeatedly cited passage of van der Made from Col. 2, line 50-Col. 3, line 15 are merely conclusory statements that are not supported by specific teachings of van der Made.

The Examiner acknowledges that van der Made does not teach detecting an intrusion into the data processing system by monitoring system calls, inspecting a stack upon detection of an intrusion to retrieve an address leading to the malicious code string, locating, as a first element on the stack, a return address of a system call entry code from which the subprogram departed, retrieving a return address of the malicious code string pointing to a memory

location in the range in which the daemon is executed from a second element on the stack positioned at or near the location of the return address of the system call entry code to facilitate finding and extracting of the malicious code string, scanning the memory range owned by the executed daemon starting from the return address in opposite directions until on one side a first region with a plurality of similar addresses and on the other side a second region with a plurality of similar instructions that do not alter the sequential control flow is identified, and extracting the malicious code string from between the first and second regions. The Examiner cites the Kilpatrick and Kolichtchak patents as teaching those claim features.

The Kilpatrick teachings from Col. 2, lines 11-22 are cited for teachings detecting an intrusion into the data processing system by monitoring system calls. Applicants concede that Kilpatrick teaches the monitoring of system calls. However, Applicants disagree with the Examiner that one having skill in the relevant art would be motivated to modify van der Made with the Kilpatrick monitoring of system calls. In order to conclude that a modification of references would have been obvious to one having skill in the art, one must find some suggestion or motivation in the

references themselves. A proper *prima facie* case of obviousness cannot be established by combining the teachings of the prior art absent some teaching, incentive, or suggestion supporting the combination. In re Napier, 55 F.3d 610, 613, 34 U.S.P.Q.2d 1782, 1784 (Fed. Cir. 1995); In re Bond, 910 F.2d 831, 834, 15 U.S.P.Q.2d 1566, 1568 (Fed. Cir. 1990).

Applicants respectfully assert that there is nothing in either van der Made or Kilpatrick that would suggest to one having skill in the art to modify van der Made with Kilpatrick. Since van der Made teaches that a program be fully executed in a virtual machine, it would not be logical to modify van der Made to run the program normally, thereby removing the protection of the virtual machine, and to monitor system calls while running the program normally. The van der Made patent expressly provides the virtual machine implementation to avoid the running of an unknown or untested program normally until a that program has been tested within the protective bubble of the virtual machine. It cannot be considered obvious to modify a reference in any way that would alter its intended use (i.e., using the virtual machine to protect the system against attacks) and/or render the reference unworkable for its intended purpose. A reference that teaches away from the claimed

invention or claimed feature thereof cannot be said to obviate that claim or claim feature (In re Gurley, 27 F.3d 551, 31 USPQ2d 1130 (Fed. Cir. 1994)). Accordingly, it would not occur to one having skill in the art to modify the protected virtual machine of van der Made with the Kilpatrick teachings to include an unprotected environment.

The Examiner further cites the Kolichtchlak patent teachings from paragraph [0032] in combination with van der Made. Kolichtchlak detects user mode programs that attempt to execute from a writable page. The fault address from which the attempt is made is compared to the execution address and if the fault address is the execution address, the process is most likely malicious code. When Kolichtchak identifies malicious code, paragraph [0032] teaches that the method logs or terminates the program that is creating the malicious code, injecting termination code in the process and changing the return address. Neither Kolichtchak the step of logging or the step of terminating is the same as or suggestive of inspecting a stack to retrieve an address leading to the malicious code string, locating a return address of a system call entry code from which the subprogram departed, retrieving a return address of the malicious code string pointing to a memory location

in the range in which the daemon is executed from a second element on the stack positioned at or near the location of the return address of the system call entry code to facilitate finding and extracting of the malicious code string, scanning the memory range owned by the executed daemon starting from the return address in opposite directions until on one side a first region with a plurality of similar addresses and on the other side a second region with a plurality of similar instructions that do not alter the sequential control flow is identified, and extracting the malicious code string from between the first and second regions. Kolichtchak does not teach locating and extracting malicious code. Rather, Kolichtchak teaches logging the attempted attack and/or terminating the program. Neither step is the same as or suggestive of the steps which are carefully enumerated in the language of independent Claims 1, 15 and 16, or the claims which depend therefrom and add further limitations thereto.

For a determination of obviousness, the prior art must teach or suggest all of the claim limitations. "All words in a claim must be considered in judging the patentability of that claim against the prior art" (In re Wilson, 424 F. 2d 1382, 1385, 165 U.S.P.Q. 494, 496 (C.C.P.A. 1970). If the cited references fail to teach each and every one of

the claim limitations, a *prima facie* case of obviousness has not been established by the Examiner. Since none of the cited references teaches the claimed steps and means for monitoring system calls to detect intrusions, followed by identifying the malicious code string related to a detected intrusion, and isolating and extracting the malicious code string as claimed, it cannot be maintained that the claims are obvious in light of the teachings of van der Made, Kilpatrick and Kolichtchak.

The Examiner has rejected Claims 4, 5, and 12 as unpatentable over Hoefelmeyer in view of Kephart. The Hoefelmeyer system is directed to a parallel scanning system for detecting malicious code. In order to save time, Hoefelmeyer's front end processor feeds the incoming communications to multiple scanning computer systems (Col. 4, line 66-Col. 5, line 1), each of which is adapted to scan communications for one or more known malicious code signatures (Col. 2, lines 62-64). When any one of the multiple scanning computer systems recognizes a malicious code signature/string, it generates an alarm.

Hoefelmeyer's system can only recognize known malicious code signatures by comparing incoming signatures/strings to stored lists of known malicious code signatures/strings. Hoefelmeyer cannot detect an intrusion

without already knowing the malicious code string. As such, Hoefelmeyer cannot detect intrusions before the intrusions occur. The Examiner states that, in Hoefelmeyer, "viruses are detected by the detection manager system". In fact, viruses are detected by the multiple scanning computer systems, 122, 124 and 126 (Col. 5, lines 15-27) which detect the viruses based on known signatures (Col. 2, lines 62-64). In a networking embodiment of the Hoefelmeyer system, a detection manager system creates a signature for each detected virus or other malicious code and transmits the new signatures to a remote scanning system.

Hoefelmeyer's system does not monitor system calls to identify intrusions. Further, Hoefelmeyer does not teach or suggest identifying a malicious code string after having detected an intrusion by monitoring system calls. Rather, Hoefelmeyer simply recognizes a predetermined code string. Finally, Hoefelmeyer does not extract the malicious code string and forward it to an intrusion limitation system. Rather, Hoefelmeyer's scanning computer system generates an alarm. Hoefelmeyer does teach that a detection management system is configured for creating a "signature" of a piece of malicious code and sending that to a remote detection location (140 of Fig. 1) when the remote location cannot

afford to run multiple computer scanning systems. What Hoefelmeyer does is send the malicious code string with identification of the type of intrusion (e.g., Trojan, etc) for the remote site to use in its monitoring/pattern matching. Hoefelmeyer is not teaching or suggesting that a signature other than the detected malicious code signature be generated nor is Hoefelmeyer teaching or suggesting isolating and extracting the malicious code before it can execute, thereby preventing damage to the host system.

The Examiner acknowledges that Hoefelmeyer does not teach correlating stored malicious code strings to find sets of malicious codes and generating a signature for each set. The Examiner cites the Kephart patent as teachings the feature missing from Hoefelmeyer. What Kephart teaches, in the cited passage from Col. 6, line 49-Col. 7, line 28, is a method which looks at a located string D for occurrences of generic features and maps occurrences of the generic features into an existing signature reference list to identify one or more known viruses having the identified signature. Kephart does not teach or suggest analyzing many code strings to group them into sets and does not teach or suggest generating a signature for a set of code strings.

The Examiner concludes that "[i]t would have been obvious at the time the invention was made to a person having ordinary skill in the art to which the subject matter pertains to have grouped...". The Applicants respectfully reiterate the argument set forth above that a rejection must be supported by teachings and that a conclusion of motivation must be supported by something other then the Examiner's conclusory statement or 20:20 hindsight. An obviousness rejection under 35 USC 103 cannot be sustained when the motivation to combine/modify references does not exist and when the references do not teach or suggest the claimed features.

Based on the foregoing arguments, Applicants respectfully request that the Examiner reconsider the obvious rejections of Claims 4, 5 and 12 based on a combination of teachings of Hoefelmeyer and Kephart, as well as the obviousness rejections of Claims 13 and 14 based on a combination of teachings of Hoefelmeyer in view of Kephart and further in view of Lamburt. With respect to Claims 13 and 14, the Examiner acknowledges that neither Hoefelmeyer nor Kephart teaches correlating by utilizing an edit distance algorithm. Applicants reiterate the argument that neither Hoefelmeyer nor Kephart teaches any correlating step to find sets of malicious code strings.

Moreover, without any teaching of a need or a step to correlate code strings to find sets, one would not be motivated to look to Lamburt for ways of correlating strings. Each of Hoefelmeyer's scanning systems simply recognizes a string and generates an alarm. There are no teachings or suggestions that multiple strings be recognized and their similarities considered. Moreover, one having skill in the art of preventing attacks on computer systems by malicious code strings would not look to the Lamburt patent teachings of on-line phone directories to modify the Hoefelmeyer or a system or Hoefelmeyer as modified by Kephart. Lamburt uses a name edit distance for matching an input name to database entries in an on-line directory. There is also nothing in Lamburt which would motivate a skilled artisan to look to Lamburt for ways to correlate malicious code strings, even if there was any correlating step taught by either Hoefelmeyer or Kephart.

Applicants again conclude that the Examiner has erred in rejecting the claims as obvious. The Federal Circuit has stated that when patentability turns on the question of obviousness, the obviousness determination "must be based on objective evidence of record" (In re Lee, 277 F. 3d 1338, 1343 (Fed. Cir. 2002)). Moreover, the Federal

Circuit has stated that "conclusory statements" by an examiner fail to adequately address the factual question of motivation, which is material to patentability and cannot be resolved "on subjective belief and unknown authority" (Id. at 1343-1344).

Claims 19 and 20 have been rejected as being unpatentable over van der Made in view of Kilpatrick and Kolichtchak and further in view of Hoefelmeyer in view of Lamburt. Applicants rely on the analysis of van der Made in view of Kilpatrick and Kolichtchak set forth above as well as the analysis of teachings from Hoefelmeyer in view of Lamburt. Since none of the cited references teaches the features of Claim 16 from which Claim 19 depends or of Claim 15, from which Claim 20 depends, since questions of motivation to combine are not answered, and since Hoefelmeyer does not teach the claimed steps or means for extracting code strings, correlating strings having mutual edit distances to assemble sets, generating signatures for the sets and distributing the signatures, it cannot be concluded that one having skill in the art would combine the references nor can it be concluded that the combination would render the claims obvious.

Applicants reiterate that a proper prima facie case of obviousness cannot be established by combining the

teachings of the prior art absent some teaching, incentive, or suggestion supporting the combination. In re Napier, 55 F.3d 610, 613, 34 U.S.P.Q.2d 1782, 1784 (Fed. Cir. 1995); In re Bond, 910 F.2d 831, 834, 15 U.S.P.Q.2d 1566, 1568 (Fed. Cir. 1990). Further, for a determination of obviousness, the prior art must teach or suggest all of the claim limitations. "All words in a claim must be considered in judging the patentability of that claim against the prior art" (In re Wilson, 424 F. 2d 1382, 1385, 165 U.S.P.Q. 494, 496 (C.C.P.A. 1970). If the cited references fail to teach each and every one of the claim limitations, a prima facie case of obviousness has not been established by the Examiner. Since none of the cited references teaches the claimed steps and means for monitoring system calls to detect intrusions, followed by identifying the malicious code string related to a detected intrusion, isolating and extracting the malicious code string, correlating code strings to find sets, generating a signature for a set of correlated code strings and forwarding the generated signature to local and remote intrusion limitation system, obviousness has not been established. The cited references do not teach the claim features and do not teach or suggest combination or modification of their respective teachings in such a way as

to arrive at the invention as claimed. Since none of the cited references teaches the claimed steps and means for monitoring system calls to detect intrusions, followed by identifying the malicious code string related to a detected intrusion, extracting the malicious code string and forwarding the malicious code string to an intrusion limitation system, obviousness has not been established.

Based on the foregoing amendments and remarks, Applicants respectfully request entry of the amendments, reconsideration of the rejections, and issuance of the claims.

Respectfully submitted,
Swimmer, et al

By: */Anne Vachon Dougherty/*
Anne Vachon Dougherty
Reg. No. 30,374
Tel. (914) 962-5910